Computer Science

Reliable Distributed Database Transactions that Require Time Sensitive Synchronization

Carl Singer (*), Jeremy Engle,Tiffany Graves,Jing Tun, Computer Science Department, DePauw University, IN 46134 runner@depauw.edu, jengle@depauw.edu, tgraves@depauw.edu, jtun@depauw.edu

Guaranteeing reliable transaction processing can be difficult in a distributed database environment that involves database servers and remote devices operating in a wireless environment. This is due to the potential unreliability of wireless connections combined with what we refer to as *blind replication* and lack of distributed transaction support.

A transaction is usually composed of several SQL statements that affect the state of a database. When a transaction completes, successfully or not, the database must be in a valid state. The database should be in a new state if the transaction is successful or in its pre-transaction state otherwise. Some database management systems provide this functionality for a distributed environment. However, in an environment that includes remote devices, such as the Pocket PC running under Windows CE distributed transaction support is missing.

Blind replication is another complicating factor in this kind of environment. One way to distribute database tables is through a process called replication. Tables or parts of tables are copied among computers in the system, providing redundancy for recovery purposes and for performance. Typically a database management system knows where tables are located and how often to synchronize them so that the database is consistent throughout the system. This is not the case for handheld computers such as the Pocket PC. All replication activity is the responsibility of the handheld computer. It determines when synchronization will take place and which tables will be synchronized. After synchronization is complete, the server providing the tables has no record that tables have been copied nor to which device. Our term for this is blind replication.

To illustrate some issues that arise when a transaction fails, consider a central database table that is replicated on a handheld computer, the handheld copy has been modified and is beginning the process of synchronizing with the central database through a wireless connection. If the connection is lost at a critical point in the process, the handheld computer may have recorded that the table has been synchronized, when in fact it never was sent because of the lost connection. Thus the distributed database is in an invalid state.

We developed a distributed database application, PocketQuiz, to help us explore issues like this and to discover other issues as well. The application allows students enrolled in classes to log in to a central database and download quizzes to a Pocket PC for one or more classes. The quiz is taken and submitted to the central database where it is recorded and graded. The central database contains information about students, faculty members, classes, and quizzes. The tables that are replicated on handheld computers describe quizzes and the students' answers. The central database resides on a Windows 2000 Server running Microsoft SQL Server. Tools used in developing the test application include Microsoft SQL Server CE and Embedded Visual Basic. The handheld platform is a collection of Compaq iPAQs with wireless cards, running Pocket PC 2002, Windows CE and SQL Server CE.

We also explored issues related to time stamp verification. For example if a quiz must be submitted by a certain time and it is, but the transaction fails, the quiz should be able to be resubmitted even though the deadline has past. How can the original time of submission be verified if handheld computer time is unreliable?